

Final Report

on

Numerical Modeling as a Tool for Analyzing Thin-Film Solar Cells and Interpreting
Device and Material Characterization Measurements

submitted to

National Renewable Energy Laboratory
1617 Cole Blvd.
Golden, Colorado 80401

Subcontract Number XAK-8-17619-36

Jeffery L. Gray
School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana 47907

Project Overview

Detailed numerical modeling has proved to be a useful tool in developing many of the current PV technologies such as single crystal Si, GaAs, $\text{?}-\text{Si}$, CdTe, and CIS, to name a few. The purpose of this work is to continue development of ADEPT (A Device Emulation Program and Tool) – primarily to integrate it with MatLabTM.

ADEPT (and/or its developmental predecessors) has been used to model single solar cells (Si, GaAs), thin-film solar cells ($\text{?}-\text{Si}$, CdTe, CIS), and multijunction solar cells. In addition, ADEPT can be used in the interpretation of device and material characterization measurements other than the basic light & dark I-V and spectral response. These include capacitance-voltage characteristics, open-circuit voltage/short-circuit current decay and lifetime measurements via dual-beam optical modulation (DBOM). Characterization measurements such as these, and their proper interpretation, are important in the development of high efficiency solar cells.

ADEPT is a detailed numerical device simulation code developed at Purdue University by the principal investigator. The so-called semiconductor equations, Poisson's equation, and the hole and electron continuity equations (assuming drift-diffusion transport), are solved numerically, subject to appropriate boundary conditions (typically an ideal ohmic contact or a Schottky-barrier contact). The equations are first discretized onto a discrete mesh and the resulting set of coupled nonlinear difference equations are solved using a generalized Newton iteration method.

Several different versions of ADEPT have been developed and have evolved somewhat independently as development proceeded for various applications to solar cell simulation. This report is a basic guide to the use the ADEPT MatLabTM Toolbox. The toolbox will provide researchers with flexibility in simulating the performance of various thin-film solar cells, as well as the simulation of device characterization measurements.

ADEPT MatLab™ Toolbox Overview

Introduction

ADEPT (and its developmental predecessors) is a detailed numerical device simulation code developed at Purdue University. The so-called semiconductor equations, Poisson's equation, and the hole and electron continuity equations (assuming drift-diffusion transport), are solved numerically, subject to appropriate boundary conditions for the contacts (typically an ideal ohmic contact or a Schottky-barrier contact) and the non-contacted surface regions (for 2D and 3D simulations). The semiconductor equations are first discretized onto a discrete mesh and the resulting set of coupled nonlinear difference equations are solved using a generalized Newton iteration method. References detailing the simulation methodology are listed in the bibliography.

Several different versions of ADEPT have been developed at Purdue and have evolved somewhat independently as development proceeded for various applications to solar cell simulation. The ADEPT MatLab™ Toolbox integrates these versions within the MatLab™ environment.

This report is a basic guide to the use the ADEPT MatLab™ Toolbox and is intended for users with a general understanding of solar cell device physics and a working knowledge of MatLab™. This toolbox will provide researchers with flexibility in simulating the performance of various thin-film solar cells, as well as the simulation of device characterization measurements.

Basic Toolbox Functions

There are six (6) basic functions that are used to perform a device simulation:

- | | |
|---------------------------|---|
| Adefine() | This function describes the device structure and discretization grid. A simulation of the device in thermal equilibrium is also performed and the equilibrium simulation result is saved for later retrieval. |
| Aset() | This function sets the non-equilibrium operating condition, i.e. the applied voltage and illumination. It can also be used to change other internal parameters. |
| Asolve() | This function performs the numerical simulation under the (presumably) new operating conditions. |
| Asave() | This function saves the result of the current simulation for later retrieval. |
| Aget() | This function retrieves the specified parameter from the current solution for use. |
| <u>Aretrieve()</u> | This function permits the retrieval of saved simulations for analysis. |

These functions are described in more detail below.

Adefine()

[*ierr,err_message*]=Adefine(*file_name*) reads the input file identified by the string *file_name*. The format of this input file is given in Appendix I. This file is used to provide a complete physical description of the device, as well as information on the construction of the discretization mesh. An output file with the name *file_name.dat* is also created and the equilibrium solution is stored therein for later retrieval and analysis. Simulation results for other operating conditions can be appended to this file with Asave().

[*ierr,err_message*]=Adefine(*file_name,itmax,dvmax,n_diverge*) reads the input file identified by the string *file_name*, as above. Additionally, the 3 parameters controlling the Newton iteration must be specified.

	default	description
<i>itmax</i>	100	maximum number of Newton iterations
<i>dvmax</i>	1×10^{-6}	convergence test value (kT)
<i>n_diverge</i>	5	# of iterations for which convergence test voltage can increase before divergence is assumed

[*ierr,err_message*]=Adefine(*file_name,'ADEPTF'*) reads the input file identified by the string *file_name*. This input file must be consistent with the input file format used in ADEPT/F, a previously released version of ADEPT. The format of this file is given in Appendix II. Only the information needed to define the device structure and discretization mesh is used. Other information is ignored.

If Adefine() executes successfully, *ierr*=0 and *err_message* is an empty string.

Aset()

[*ierr,err_message*]=Aset('V_DIODE',*applied_voltage*) sets the applied voltage at all p-type contacts to the value *applied_voltage*, which is specified in Volts. The n-type contacts are kept at their equilibrium potential. Thus, for a diode device (such as a solar cell), a for

[*ierr,err_message*]=Aset('V_CONTACT',*contact_number,applied_voltage*) sets the applied voltage at the contact identified by *contact_number* to the value *applied_voltage*, which is specified in Volts.

[*ierr,err_message*]=Aset('OPEN-CIRCUIT') sets the boundary conditions to correspond to an open-circuit condition.

`[ierr,err_message]=Aset('DARK')` sets the optical generation rate everywhere inside the device to zero.

`[ierr,err_message]=Aset('UNIFORM_GEN',gen_rate)` sets the optical generation rate everywhere inside the device to gen_rate ($\text{cm}^{-3}\cdot\text{s}^{-1}$).

`[ierr,err_message]=Aset('MONOCHROMATIC',Jinc,wl)` sets the illumination to monochromatic light of wavelength wl (μm) with and incident intensity of $q \times Jinc$ (Amperes/cm²). The light is assumed to be incident at $x=0$ normal to the $y-z$ plane propagating in the positive x direction.

`[ierr,err_message]=Aset('SPECTRUM',file,conc)` sets the incident illumination to the spectrum defined in `file`. The value of `conc` is used to set the light intensity relative to that defined in the spectrum file. The light is assumed to be incident at $x=0$ normal to the $y-z$ plane propagating in the positive x direction.

If `Aset()` executes successfully, `ierr=0` and `err_message` is an empty string.

Asolve()

`[ierr,err_message]=Asolve('DC')` performs the numerical simulation at steady-state under the current operating conditions using the default matrix solution technique – LU decomposition of a banded matrix.

`[ierr,err_message]=Asolve('DC','LU-BANDED')` performs the numerical simulation under the current operating conditions using LU decomposition of a banded matrix, the default method.

`[ierr,err_message]=Asolve('DC',method,itmax,dvmax,n_diverge)` For both cases above, the 3 parameters controlling the Newton iteration can be specified.

	default	description
<code>itmax</code>	100	maximum number of Newton iterations
<code>dvmax</code>	1×10^{-6}	convergence test value (kT)
<code>n_diverge</code>	5	# of iterations for which convergence test voltage can increase before divergence is assumed

If `Asolve()` executes successfully, `ierr=0` and `err_message` is an empty string.

Asave()

[*ierr,err_message*]=Asave(*save_message*) appends the results of the current simulation solution to *file_name.dat* (see Adefine()).

If Asave() executes successfully, *ierr*=0 and *err_message* is an empty string.

Aget()

[*current,ierr,err_message*]=Aget('CURRENT','DIODE') returns the value of the current (A/cm^2) flowing out of the p-type contacts.

[*current,ierr,err_message*]=Aget('CURRENT','CONTACT',*contact_number*) returns the value of the current (A/cm^2) flowing out of the contact identified by *contact_number*.

[*xyz,value,ierr,err_message*]=Aget(*param*) returns the an array value of the specified parameter at each node of the discretization mesh. *xyz* is a 1-, 2-, or 3-dimesional array of the spatial coordinates (cm) depending on whether a 1D, 2D, or 3D simulation was performed (3D simulation is currently not supported). The following parameters may be requested: 'p' (hole concentration, cm^{-3}); 'n' (electron concentration, cm^{-3}); 'V' (electrostatic potential, Volts); 'R' (recombination rate, $\#\text{/cm}^3\text{-s}$); 'G' (generation rate, $\#\text{/cm}^3\text{-s}$), 'EG' (band gap, eV); 'CHI' (electron affinity, eV); and 'N' (net impurity density, cm^{-3}).

If Aget () executes successfully, *ierr*=0 and *err_message* is an empty string.

Aretrieve()

The following function permits the retrieval of saved simulations for analysis.

[*ierr,err_message*]=Aretrieve(*file_name*) when called for the first time, retrieves the first set of simulation data from the file *file_name.dat*. Each subsequent call retrieves the next set simulation data from the file. *ierr*=-1 when the end of the file (EOF) is reached. If Aretrieve() executes successfully, *ierr*=0 and *err_message* is an empty string.

Sample Script to Perform a Simulation

Once input files have been created (see Appendices I and II), MatLab™ can be used to perform device simulations. Suppose the following ADEPT/F input file, 'test' has been created.

```
*title CIS device, based on JAP paper by Schmid, et. al., Stuttgart
      JAP 73(6) 15 March 1993, pp.2902-2909
misc    tempc=25.0
mesh    nx=250 wt=1/1/2/1 xres=1
```

```

*layer n+ ZnO -- Eg and Nc adjusted for degeneracy, Ev cont.
layer tm=0.45 eg=3.05 chi=4.13 ks=8.5 nc=4.383e15 nv=6.067e19
+
nd=1e20 ead=-1 up=1 un=8
+
taup.shr=1e-8 taun.shr=1e-8 c0.egl=1e5
*layer n ZnO
layer tm=0.05 eg=3.3 chi=3.88 ks=8.5 nc=4.1285e18 nv=6.067e19
+
nd=1e15 ead=-1 up=2 un=16
+
taup.shr=1e-8 taun.shr=1e-8 c0.egl=1e5
*layer CdS
layer tm=.02 eg=2.42 chi=4.03 ks=10 nc=1.6e18 nv=1.8e19
+
nd=1e15 ead=-1 up=10 un=20
+
taup.shr=3e-7 taun.shr=3e-7 egl.opt=2.42 c0.egl=1.8e5
*layer OVC layer
layer tm=.1 eg=1.3 chi=4.29 ks=13.6 nc=6.6e17 nv=1.5e19
+
nd=3e12 ead=-1 up=25 un=50
+
taup.shr=2e-9 taun.shr=2e-9 egl.opt=1.3 c0.egl=1e5
*layer CIS
layer tm=2.0 model=cis.st
+
eg=1.04 chi=4.0
+
na=2e16 eaa=-1 up=50 un=100
genrec gen=am1p5gn
solcell vstart=0
solve itmax=100 delmax=1.e-6

```

The objective is to duplicate the simulation results within the MatLab™ environment. The following MatLab™ script will accomplish this. Note that the genrec, solcell, and solve input lines will be ignored and their instructions must therefore be incorporated into the MatLab™ script.

```

% solve at short-circ uit, get current, and save
v(1)=0.0
[ierr,errmess]=Asolve('DC')
[j(1),ierr,errmess]=Aget('CURRENT','DIODE')
comment=sprintf('AM1.5gn spectrum: V= %f J=%f',v(1),j(1));
[ierr,errmess]=Asave(comment)

% compute i-v up to max power point and save
stop=0;
i=1;
prev_power=0
while stop == 0
    i=i+1;
    v(i)=v(i-1)+0.1;
    [ierr,errmess]=Aset('V_DIODE',v(i))
    [ierr,errmess]=Asolve('DC')
    [j(i),ierr,errmess]=Aget('CURRENT','DIODE')
    power=v(i)*j(i);
    if power > prev_power
        comment=sprintf('AM1.5gn spectrum: V= %f J=%f',v(i),j(i));
        [ierr,errmess]=Asave(comment)
        prev_power=power;
    else
        stop=1;
    end
end

% compute i-v up to open-circuit and save
stop=0;
i=i-1;
while stop == 0

```

```

i=i+1;
v(i)=v(i-1)+0.02;
[ierr,errmess]=Aset('V_DIODE',v(i))
[ierr,errmess]=Asolve('DC')
[j(i),ierr,errmess]=Aget('CURRENT','DIODE')
if j(i) > 0
    comment=sprintf('AM1.5gn spectrum: V= %f J=%f',v(i),j(i));
    [ierr,errmess]=Asave(comment)
else
    % compute open-circuit and save
    [ierr,errmess]=Aset('OPEN-CIRCUIT')
    [ierr,errmess]=Asolve('DC')
    [v(i),ierr,errmess]=Aget('VOLTAGE','DIODE')
    [j(i),ierr,errmess]=Aget('CURRENT','DIODE')
    comment=sprintf('AM1.5gn spectrum: V= %f J=%f',v(i),j(i));
    [ierr,errmess]=Asave(comment)
    stop=1;
end
end

```

This demonstrates the relative ease of execution of device simulation that is attained by integrating ADEPT into the MatLab™ environment. Scripts to simulate various device characteristics can be created and customized for specific analyses. Similar scripts for other analyses, such as dark I-V and spectral response, can be similarly constructed.

Interpreting the Simulation Results

MatLab™ presents the opportunity for interactive interpretation of the simulation results. Results from specific simulations can be retrieved using Aretrieve(). Specific parameters can then be extracted with Aget() and then plotted using the graphical functions available in MatLab™. While the built in graphics functions provide adequate ability to display the simulation results, future releases of the ADEPT MatLab™ Toolbox will include a library of plotting routines tailored to this application.

Summary

This document has described the key functions available in the ADEPT MAtLab™ Toolbox and provides basic instruction of their use. The tools available in the toolbox will be expanded in future releases. This includes the library of plotting routines mentioned above, a library of customized scripts for various device and materials characterization experiments, more robust 2D simulations, and the ability to perform robust 3D simulations.

Bibliography

1. M. S. Lundstrom*, R. J. Schwartz*, J. L. Gray*, "Transport Equations for the Analysis of Heavily Doped Semiconductor Devices," Solid-State Electronics, Vol. 24, pp. 195-202, 1981.
2. J. L. Gray* and M. S. Lundstrom*, "Numerical Solution of Poisson's Equations with Application to C-V Analysis of III-V Heterojunction Capacitors," IEEE Trans. Electron Devices, Vol. ED-32, pp. 2102-2109, 1985.
3. J. L. Gray, "Computer Simulation of Thin Film Silicon-Hydrogen Alloy Solar Cells," IEEE Trans. Electron Devices, Vol. 36, No. 5, pp. 906-912, 1989.
4. S. M. Durbin* and J. L. Gray*, "Numerical Modeling of Photon Recycling in Solar Cells", IEEE Trans. Electron Devices, Vol. 41, No. 2, pp. 239-245, 1994.
5. Edward H. S. Hsing and Jeffery L. Gray, "Numerical Analysis of Relationship Between Block and Driving Capability in 6H-SiC UMOSFET," Microelectronics Journal **30**, pp. 1-14, 1999.
6. M. S. Lundstrom*, R. J. Schwartz*, and J. L. Gray*, "Modeling Solar Cells Containing Heavily Doped Regions," Conference Record of the Fifteenth IEEE Photovoltaic Specialists Conference, pp. 400-405, Orlando, Florida, 1981.
7. M. S. Lundstrom*, J. L. Gray*, and R. J. Schwartz*, "Computer Simulation of Silicon Solar Cells," Proceedings of the NASECODE II Conference, pp. 223-227, Dublin, Ireland, June 17-19, 1981.
8. J. L. Gray*, R. J. Schwartz*, M. S. Lundstrom*, and R. D. Nasby*, "The Intensity Dependence of Surface Recombination in High Concentration Solar Cells with Charge Induced Passivation," Conference Record of the Sixteenth IEEE Photovoltaic Specialists Conference, pp. 437-441, San Diego, California, 1982.
9. J. L. Gray*, R. J. Schwartz*, and R. D. Nasby*, "Two-Dimensional Effects in Conventional Solar Cells Operated at High Intensities," Proceedings of the International Electron Devices Meeting, pp. 510-513, San Francisco, California, 1982.
10. J. L. Gray* and R. J. Schwartz*, "Two-Dimensional Computer Simulation of the EMVJ and Grating Solar Cells Under AMO Illumination," Proceedings of the Space Photovoltaic Research and Technology Conference, pp. 34-40, NASA Lewis Research Center, Cleveland, Ohio, Oct. 18-20, 1983.

11. R. J. Schwartz*, J. L. Gray*, G. B. Turner*, D. Kanani*, H. Ullal*, "Numerical Modeling of P-I-N Hydrogenated Thin Film Silicon Solar Cells," Conference Record of the Seventeenth IEEE Photovoltaic Specialists Conference, pp. 369-373, Kissimmee, Florida, May 1-4, 1984.
12. J. L. Gray* and R. J. Schwartz*, "Two-Dimensional Computer Simulation of Single Crystal Silicon Concentrator Cells," Conference Record of the Seventeenth IEEE Photovoltaic Specialists Conference, pp. 1297-1302, Kissimmee, Florida, May 1-4, 1984.
13. R. J. Schwartz*, J. L. Gray*, and M. S. Lundstrom*, "Current Status of One- and Two-Dimensional Numerical Models: Successes and Limitations," presented at the High Efficiency Crystalline Silicon Solar Cell Research Forum, Phoenix, Arizona, July 9-11, 1984.
14. R. J. Schwartz*, J. L. Gray*, and G. B. Turner*, "P-I-N Thin Film Silicon Hydrogen Alloy Solar Cells: Numerical Model Predictions," Proceedings of the First International Photovoltaic Science and Engineering Conference, pp. 123-126, Kobe, Japan, Nov. 13-16, 1984.
15. G. B. Turner*, K. W. Mitchell*, R. J. Schwartz*, and J. L. Gray*, "P-I-N Thin Film Silicon Hydrogen Alloy Solar Cells: Comparison with Experiment," Proceedings of the First International Photovoltaic Science and Engineering Conference, pp. 679-682, Kobe, Japan, Nov. 13-16, 1984.
16. M. S. Lundstrom* and J. L. Gray*, "Solution of Poisson's Equation in III-V Heterostructures," presented at the Second SIAM-IEEE Conference on Numerical Simulation of VLSI Devices, Boston, Mass., Nov. 12, 1984.
17. G. B. Turner*, R. J. Schwartz*, and J. L. Gray*, "Thin Film Silicon Hydrogen Alloy Solar Cells: Performance Dependence on Design and Materials Parameters," Proceedings of the Sixth European Photovoltaic Solar Energy Conference, pp. 1-8, London, U.K., April 15-19, 1985.
18. J. L. Gray* and R. J. Schwartz*, "Why Don't We Have a 30% Efficient Silicon Solar Cell?", Conference Record of the Eighteenth IEEE Photovoltaic Specialists Conference, pp. 568-572, Las Vegas, Nevada, Oct. 21-25, 1985.
19. R. J. Schwartz*, G. B. Turner*, J. W. Park*, J. L. Gray*, "Numerical Modeling of Thin Film SiH Solar Cells," SPIE Vol. 763 - *Physics of Amorphous Semiconductor Devices*, pp. 126-133, Los Angeles, California, Jan. 16, 1987.
20. J. W. Park*, R. J. Schwartz*, J. L. Gray*, and G. B. Turner*, "A Self-Consistent Numerical Model of Thin Film Silicon-Hydrogen Alloy Solar Cells," Conference Record of the Twentieth IEEE Photovoltaic Specialists Conference, pp. 55-60, Las Vegas Nevada, September 27-30, 1988.

21. G. B. Turner*, R. J. Schwartz*, and J. L. Gray*, "Band Discontinuity and Bulk vs. Interface Recombination in CdS/CuInSe₂ Solar Cells," Conference Record of the Twentieth IEEE Photovoltaic Specialists Conference, pp. 1457-1460, Las Vegas Nevada, September 27-30, 1988.
22. R. J. Schwartz*, J. W. Park*, J. L. Gray*, and G. B. Turner*, "Numerical Modeling of a-Si:H Thin Film Solar Cells," PVSEC-4, Australia, Feb. 13-17, 1989.
23. J. L. Gray, "Numerical Model for Tunnel Junctions with Application to Cascade Solar Cells," Conference Record of the Twenty-First IEEE Photovoltaic Specialists Conference, pp. 345-347, Kissimmee, Florida, May 1990.
24. R. J. Schwartz*, J. L. Gray*, "The Use of CuIn_{1-x}Ga_xSe₂ Layers to Improve the Performance of CuInSe₂ Cells," Conference Record of the Twenty-First IEEE Photovoltaic Specialists Conference, pp. 570-574, Kissimmee, Florida, May 1990.
25. J. L. Gray, "ADEPT: A General Purpose Numerical Device Simulator for Modeling Solar Cells in One-, Two-, and Three-Dimensions," Conference Record of the Twenty-Second IEEE Photovoltaic Specialists Conference, pp. 436-438, Las Vegas, Nevada, October 1991.
26. J. L. Gray* and Y. J. Lee*, "Numerical Modeling of CdS/CdTe Solar Cells: A Parameter Study," Conference Record of the Twenty-Second IEEE Photovoltaic Specialists Conference, pp. 1151-1155, Las Vegas, Nevada, October 1991.
27. S. M. Durbin* and J. L. Gray*, "Numerical Modeling of Photon Recycling in High Efficiency GaAs Solar Cells," Conference Record of the Twenty-Second IEEE Photovoltaic Specialists Conference, pp. 188-191, Las Vegas, Nevada, October 1991.
28. R. J. Schwartz*, J. L. Gray*, and Y. J. Lee*, "Design Considerations for Thin Film CIS and Other Polycrystalline Heterojunction Solar Cells," Conference Record of the Twenty-Second IEEE Photovoltaic Specialists Conference, pp. 920-923, Las Vegas, Nevada, October 1991.
29. J. L. Gray*, R. J. Schwartz*, and Y. J. Lee*, "Numerical Modeling of Polycrystalline Thin Film Solar Cells," AIP Conference Proceedings 268, Photovoltaic Advanced Research and Development Project, Denver, Colorado, pp. 102-107, 1992.
30. M. S. Lundstrom*, M. R. Melloch*, G. B. Lush*, M. P. Patkar*, M. Young*, S. M. Durbin*, and J. L. Gray, "Radiative Recombination and Photon Recycling in Gallium Arsenide Solar Cells," AIP Conference Proceedings 268, Photovoltaic Advanced Research and Development Project, Denver, CO, pp. 298-303, 1992.

31. S. M. Durbin, J. L. Gray, and R. K. Ahrenkiel, "Numerical Modeling of the Influence of Photon Recycling on Lifetime Measurements", Conference Record of the 23rd IEEE Photovoltaic Specialists Conference, Louisville, Kentucky, May 10-14, 1993.
32. S. M. Durbin and J. L. Gray, "Difficulties Encountered in Developing Numerical InP Solar Cell Models," Conference Record of the 23rd IEEE Photovoltaic Specialists Conference, Louisville, Kentucky, May 10-14, 1993.
33. Y. J. Lee and J. L. Gray, "Numerical Modeling of Polycrystalline CdTe and CIS Solar Cells", Conference Record of the 23rd IEEE Photovoltaic Specialists Conference, Louisville, Kentucky, May 10-14, 1993.
34. J. L. Gray and Y. J. Lee, "Modeling of the Temperature Dependent Behavior of the Molybdenum/CIS Contact in CIS-Based Solar Cells", AIP Conference Proceedings 306, 12th NREL Photovoltaic Program Review, Denver Colorado, pp. 305-308, October 1993.
35. [43] S. M. Durbin and J. L. Gray, "A Detailed Multi-Dimensional Tool for Analyzing InP Solar Cells", Proceedings of the 12th EC Photovoltaic Solar Energy Conference, pp. 1323-1326, April 1994, Amsterdam, The Netherlands.
36. Y-J. Lee and J. L. Gray, "Numerical Modeling of the Temperature and Illumination Dependent Performance of CIS Solar Cells", Proceedings of the 12th EC Photovoltaic Solar Energy Conference, pp. 1561-1563, April 1994, Amsterdam.
37. S. M. Durbin, D. H. Levi, and J. L. Gray, "A Numerical Model for Computing the Emission Spectrum in Time -Resolved Photoluminescence Experiments", presented at the 3rd International Workshop on Computational Electronics, Portland, Oregon, May 18-20, 1994.
38. J. L. Gray and Y. J. Lee, "Numerical Modeling of Graded Band Gap CIGS Solar Cells", Proceedings of at the First World Conference on Photovoltaic Energy Conversion, pp. 123-126, Waikoloa, Hawaii, December 5-9, 1994.
39. Y. J. Lee and J. L. Gray, "The Effects of Band Bending Caused by Interface States in CdTe/CIS Solar Cells", Proceedings of the First World Conference on Photovoltaic Energy Conversion, pp 287-290, Waikoloa, Hawaii, December 5-9, 1994.
40. J. L. Gray, "Interpretation of Capacitance-Voltage Characteristics in Thin-Film Solar Cells Using a Detailed Numerical Model", Conference Record of the Twenty-Fifth IEEE Photovoltaic Specialists Conference, pp. 905-908, Washington, D.C., May 13-17, 1996.

41. J. L. Gray and R. J. Schwartz, "Numerical Modeling of III-V Photovoltaic Devices", Proceedings of the 9th International Photovoltaic Science and Engineering Conference, Miyazaki, Japan, November 11-15, 1996.
42. J. L. Gray and Thomas K. Simacek, "Numerical Modeling of Thin Film Solar Cells", presented at the NREL/SNL Photovoltaics Program Review Meeting, Lakewood, Colorado, November 18-22, 1996.
43. Thomas K. Simacek and Jeffery L. Gray, "Interpretation of Dual Beam Optical Modulation (DBOM) Measurements in Thin-Film Materials", Conference Record of the Twenty-Sixth IEEE Photovoltaic Specialists Conference, pp. 475-478, Anaheim, California, September 29 - October 3, 1997.

Appendix I

This contains information on the input file format for the ADEPT MatLab™ Toolbox.
Shown are the C routines (callable from MatLab™) that read the input file.

```

#include "../A_dir.h"
#include DIRECTORY/MAIN/A_config.h"
#include DIRECTORY/MAIN/A_adept.h"
#include DIRECTORY/MODELS/A_models.h"
#include DIRECTORY/MODELS/A_cntct.h"
#include DIRECTORY/MODELS/A_ntrfc.h"
#include DIRECTORY/MESH/A_domain.h"
#include DIRECTORY/MAIN/A_normal.h"

input_regions()
{
    int i,j,c;
    char file[50];

    suffix(root,".reg\0",file);
    printf("regfile = %s\n",file);
    if((regfile=fopen(file,"r"))==0){
        printf("error opening %s\n",file);
        printf("root = %s\n",root);
        exit(1);
    }

    fscanf(regfile,"%lf",&ref.t);
    fprintf(report,"temperature = %e Kelvin\n",ref.t);

    setnorref();

    fscanf(regfile,"%d",&num_reg);
    fprintf(report,"# regions = %d\n",num_reg);

    for(i=0;i<num_reg;i++){

        fprintf(report,"\nregion %d:\n",i);

        fgets(reg[i].describe,80,regfile); /* get by NEWLINE in previous
line */
        fgets(reg[i].describe,80,regfile);
        fprintf(report,"%s\n",reg[i].describe);
        fflush(report);

        for(j=0;j<DIM;j++){
            fscanf(regfile,"%lf %lf",&reg[i].min[j],&reg[i].max[j]);
            fprintf(report,"%d min = %e max = %e\n",j,reg[i].min[j],reg[i].max
[j]);
        }
        fscanf(regfile,"%d",&reg[i].type);
        fprintf(report,"region type = %d",reg[i].type);
        fscanf(regfile,"%d",&reg[i].mat);
        fprintf(report,"region mat = %d\n",reg[i].mat);
        switch(reg[i].type){
            case 0:
                input_insul(i);
                break;

```

```

    case 1:
        input_semi(i);
        break;
    default:
        printf("undefined region type\n");
        fprintf(debug,"undefined region type\n");
        exit(1);
    }
}

fscanf(regfile,"%d",&num_contact);
fprintf(report,"\n# contacts = %d\n",num_contact);

for(i=0;i<num_contact;i++){
    fscanf(regfile,"%d",&contact[i].type);
    fprintf(report,"contact %d: type = %d\n",i,contact[i].type);
    for(j=0;j<DIM;j++){
        fscanf(regfile,"%lf %lf",&contact[i].min[j],&contact[i].max[j]);
        fprintf(report,"%d cmin = %e cmax = %e\n",j,contact[i].min
[j],contact[i].max[j]);
    }
    switch(contact[i].type){
        case 0: /* no additional input for this contact type */
            break;
        case 1: /* 1D only, read in Sp, Sn */
            if(DIM==1){
                fscanf(regfile,"%lf %lf",&contact[i].type_1.sp,&contact
[i].type_1.sn);
            }else{
                printf("type 1 defined for 1D only\n");
                fprintf(debug,"type 1 defined for 1D only\n");
                exit(1);
            }
            break;
        default:
            printf("undefined contact type\n");
            fprintf(debug,"undefined contact type\n");
            exit(1);
    }
}

fscanf(regfile,"%d",&num_interface);
fprintf(report,"\n# interfaces = %d\n",num_interface);

for(i=0;i<num_interface;i++){
    fscanf(regfile,"%d",&interface[i].face_node_reg);
    for(j=0;j<DIM;j++){
        fscanf(regfile,"%lf %lf",&interface[i].min[j],&interface[i].max
[j]);
        fprintf(report,"%d imin = %e imax = %e\n",j,interface[i].min
[j],interface[i].max[j]);
    }
    switch(reg[interface[i].face_node_reg].mat){

```

```

    case 3: /* silicon model */
        fscanf(regfile,"%lf",&interface[i].sil_3.qss);
        fprintf(report,"qss = %e\n",interface[i].sil_3.qss);
        fscanf(regfile,"%lf",&interface[i].sil_3.sp);
        fprintf(report,"sp = %e\n",interface[i].sil_3.sp);
        fscanf(regfile,"%lf",&interface[i].sil_3.sn);
        fprintf(report,"sn = %e\n",interface[i].sil_3.sn);
        fscanf(regfile,"%lf",&interface[i].sil_3.ets);
        fprintf(report,"ets = %e\n",interface[i].sil_3.ets);
        break;
    case 4: /* algaas model */
        fscanf(regfile,"%lf",&interface[i].gal_4.qss);
        fprintf(report,"qss = %e\n",interface[i].gal_4.qss);
        fscanf(regfile,"%lf",&interface[i].gal_4.sp);
        fprintf(report,"sp = %e\n",interface[i].gal_4.sp);
        fscanf(regfile,"%lf",&interface[i].gal_4.sn);
        fprintf(report,"sn = %e\n",interface[i].gal_4.sn);
        fscanf(regfile,"%lf",&interface[i].gal_4.ets);
        fprintf(report,"ets = %e\n",interface[i].gal_4.ets);
        break;
    default:
        fprintf(report,"no interface model for mat=%d\n",
            reg[interface[i].face_node_reg].mat);
    }
}

```

```

#include "../A_dir.h"
#include DIRECTORY/MAIN/A_config.h"
#include DIRECTORY/MAIN/A_adept.h"
#include DIRECTORY/MESH/A_mshgen.h"
#include DIRECTORY/MESH/A_domain.h"

input_mesh()
{
    void con_ele();
#if ONE_DIM == ON
    int i,nx,flag;
    double x[MAX_NODES];
    char file[50];

    suffix(root,".msh\0",file);
    printf("mshfile = %s\n",file);
    if((mshfile=fopen(file,"r"))==0){
        printf("error opening %s\n",file);
        printf("root = %s\n",root);
        exit(1);
    }
    fscanf(mshfile,"%d",&flag);

    switch(flag){
        case 0:
            fscanf(mshfile,"%d",&nx);
            fprintf(debug,"\nnx = %d\n",nx);
            for(i=0;i<nx;i++){
                fscanf(mshfile,"%lf",&x[i]);
                fprintf(debug,"x = %e\n",x[i]);
            }
            def_grid_1d(nx,x);
            break;
        case 1:
            fscanf(mshfile,"%d",&autol->nxf);
            fprintf(debug,"\nnx = %d\n",autol->nxf);
            fscanf(mshfile,"%d",&autol->ismth1);
            fprintf(debug,"ismth1 = %d\n",autol->ismth1);
            for(i=0;i<4;i++){
                fscanf(mshfile,"%lf",&autol->cc[i]);
                fprintf(debug,"wt= %e ",autol->cc[i]);
            }
            fprintf(debug,"\n");
            autol->num_layer=num_reg;
            for(i=0;i<num_reg;i++){
                fscanf(mshfile,"%lf",&autol->xres[i]);
                fprintf(debug,"xres = %e\n",autol->xres[i]);
            }
            auto_mesh_1d(x);
            def_grid_1d(autol->nxf,x);
            break;
        default:
            fprintf(debug,"Illegal mesh option \n",flag);
    }
}

```

```

        exit(1);
    }
#endif
#if TWO_DIM_REC == ON
    int i,nx,ny,flag;
    double x[MAX_X];
    double y[MAX_Y];
    char file[50];

    suffix(root,".msh\0",file);
    printf("mshfile = %s\n",file);
    if((mshfile=fopen(file,"r"))==0){
        printf("error opening %s\n",file);
        printf("root = %s\n",root);
        exit(1);
    }
    fscanf(mshfile,"%d",&flag);

    switch(flag){
        case 0:
            fscanf(mshfile,"%d",&nx);
            fprintf(debug,"\nnx = %d\n",nx);
            for(i=0;i<nx;i++){
                fscanf(mshfile,"%lf",&x[i]);
                fprintf(debug,"x = %e\n",x[i]);
            }
            fscanf(mshfile,"%d",&ny);
            fprintf(debug,"\nny = %d\n",ny);
            for(i=0;i<ny;i++){
                fscanf(mshfile,"%lf",&y[i]);
                fprintf(debug,"y = %e\n",y[i]);
            }

            def_grid_2dr(nx,x,ny,y);
            break;
        default:
            fprintf(debug,"Illegal mesh option \n",flag);
            exit(1);
    }
/* set up stuff for 2D generation */

    set_2dr_gparams(x,nx,y,ny);

#endif PLOT == ON
    suffix(root,".grd\0",file);
    plots(5,0,file);
    plot(.1,.1,-3);
    factor(0.8);
    plot_grid_2d();
    plot(0.,0.,999);
#endif
#endif
#endif THREE_DIM_REC == ON

```

```

int i,nx,ny,nz,flag;
double x[MAX_X];
double y[MAX_Y];
double z[MAX_Z];
char file[50];

suffix(root,".msh\0",file);
printf("mshfile = %s\n",file);
if((mshfile=fopen(file,"r"))==0){
    printf("error opening %s\n",file);
    printf("root = %s\n",root);
    exit(1);
}
fscanf(mshfile,"%d",&flag);

switch(flag){
    case 0:
        fscanf(mshfile,"%d",&nx);
        fprintf(debug,"\nnx = %d\n",nx);
        for(i=0;i<nx;i++){
            fscanf(mshfile,"%lf",&x[i]);
            fprintf(debug,"x = %e\n",x[i]);
        }
        fscanf(mshfile,"%d",&ny);
        fprintf(debug,"\nny = %d\n",ny);
        for(i=0;i<ny;i++){
            fscanf(mshfile,"%lf",&y[i]);
            fprintf(debug,"y = %e\n",y[i]);
        }
        fscanf(mshfile,"%d",&nz);
        fprintf(debug,"\nnz = %d\n",nz);
        for(i=0;i<nz;i++){
            fscanf(mshfile,"%lf",&z[i]);
            fprintf(debug,"z = %e\n",z[i]);
        }
        def_grid_3dr(nx,x,ny,y,nz,z);
        break;
    default:
        fprintf(debug,"Illegal mesh option \n",flag);
        exit(1);
}

#if PLOT == ON
suffix(root,".grd\0",file);
plots(5,0,file);
plot(.1,.1,-3);
factor(0.8);
plot_grid_3d();
plot(0.,0.,999);
#endif
#endif

con_ele();

```


Appendix II

This contains information on the input file format from ADEPT/F for use with the ADEPT MatLab™ Toolbox.

*TITLE

All characters following *TITLE (up to column 80) become part of a comment used to annotate the output.

SOLVE

Parameter	Type	Default	Description
ITMAX	integer	100	maximum # of Newton iterations
DELMAX	real	1×10^{-6}	convergence test (kT)
MAXDEL	real	1×10^{50}	
NDVRGE	integer	5	# of iterations for which ΔV can increase before divergence is assumed
NBACKUP	integer	3	
NDVBIG	integer	2	
METHOD	integer	1	
TAU	real	1.0	for METHOD=2

OUTPUT

Parameter	Type	Default	Description
INFO	integer	3	level of output 0 for minimum 5 for debugging
STEP	integer	5	for tables, print values every STEP nodes
COPIES	integer	2	# copies of summary info

MESH

Parameter	Type	Default	Description
NX	integer	250	total # of nodes in F-D mesh
XRES	real	0.5	min. spatial resolution (angstroms)
MFLAG	integer	0	if =0, initial mesh is redefined based on refinement criteria described below if =1, mesh is not redefined
WT	real()		weighting parameters for mesh refinement 1.0 WT(1), ρ slowly varying 1.0 WT(2), \mathbf{E} slowly varying 0.5 WT(3), tendency for uniform mesh 0.5 WT(4), tendency for same # nodes in each LAYER
ISMTH0	integer	0	# of nodes set aside (per LAYER) for smoothing of redefined mesh
ISMTH1	integer	6	# of nodes set aside (per LAYER) for smoothing of initial mesh
XFIX	real()	-1.	position(s) at which a node must be placed (angstroms) to enable, set XFIX(1)=0.0
NNX	integer()	-1	# nodes to be placed between fixed node positions (positioning of these nodes is still based on redefinition scheme)

MISC

Parameter	Type	Default	Description
TEMPC	real	27.0	temperature ($^{\circ}\text{C}$)
TEMPK	real	300.15	temperature (Kelvin)
VTH(300)	real	1×10^7	thermal velocity at 300 K (cm/s)

BC

Parameter	Type	Default	Description
MBC	integer	1	sets contact boundary conditions =1 for charge neutral contacts =2, then electrostatic potential at contacts set by metal work functions and semiconductor electron affinity, ϕ_{ms} =3 for front neutral, back ϕ_{ms} =4 for back neutral, front ϕ_{ms}
WDFRT	real	0.86	$\phi_{\text{M}} - \chi_{\text{S}}$ at front contact (eV)
WDBCK	real	0.86	$\phi_{\text{M}} - \chi_{\text{S}}$ at back contact (eV)
SPF	real	-1.0	surface recombination velocity for holes at $x=0$ (cm/s) (=-1 sets SPF= ∞)
SNF	real	-1.0	surface recombination velocity for electrons at $x=0$ (cm/s)
SPB	real	-1.0	surface recombination velocity for holes at $x=x_{\text{max}}$ (cm/s)
SNB	real	-1.0	surface recombination velocity for electrons at $x=x_{\text{max}}$ (cm/s)

GLOBAL

This card must appear before any LAYER card. The default value for each variable on the LAYER card can be changed on this card (except T, the layer thickness).

*LAYER

All characters following *LAYER (up to column 75) become part of a comment used to annotate the output. It is intended to provide additional descriptive information about the associated LAYER.

LAYER

Parameter	Type	Default	Description
T	real		thickness of layer (angstroms)
TM	real		thickness of layer (microns)
EG EGL EGR	real	1.0	bandgap (eV)
CHI CHIL CHIR	real	4.0	electron affinity (eV)
KS	real	1.0	dielectric constant
NDX	real	1.0	index of refraction
NV NVL NVR	real	1×10^{20}	valence band effective density of states (cm^{-3})
NC NCL NCR	real	1×10^{20}	conduction band effective density of states (cm^{-3})
UP UPL UPR VSATP BBP	real real	1.0 2.0	hole mobility ($\text{cm}^2/\text{V}\cdot\text{s}$) saturation vel. for holes (cm/sec) power law for field dep mobility
UN UNL UNR VSATN BBN	real real	1.0 2.0	electron mobility ($\text{cm}^2/\text{V}\cdot\text{s}$) saturation vel. for electrons (cm/sec) power law for field dep mobility

LAYER (continued)

Parameter	Type	Default	Description
PROF.NA	char	LINEAR	doping profile type (only option at present)
NA NAL NAR	real real real	0.0	acceptor dopant density (cm^{-3})
NA.SIG	real	0.0	standard deviation (in eV) of gaussian distribution of acceptors
GA	real	1.0	acceptor degeneracy factor
EAA EAAL EAAR	real	0.0	acceptor energy level (eV above E_V)
XNA	real	0.0	(see notes) (angstroms)
PROF.ND	char	LINEAR	doping profile type (only option at present)
ND NDL NDR	real real real	0.0	donor dopant density (cm^{-3})
ND.SIG	real	0.0	standard deviation (in eV) of gaussian distribution of donors
GD	real	1.0	donor degeneracy factor
EAD EADL EADR	real	0.0	donor energy level (eV below E_C)
XND	real	0.0	(see notes) (angstroms)

LAYER (continued)

Parameter	Type	Default	Description
KTA.T	real	.001	characteristic energy of conduction band tail (eV)
KTD.T	real	.001	characteristic energy of valence band tail (eV)
GAMAX.T	real	0.0	density of states of conduction band tail at E_C ($\text{cm}^{-3}\text{-eV}^{-1}$) (-1. sets GAMAX.T = NC/(1eV)) (states assumed to be acceptor-like)
GDMAX.T	real	0.0	density of states of valence band tail at E_V ($\text{cm}^{-3}\text{-eV}^{-1}$) (-1. sets GDMAX.T = NV/(1eV)) (states assumed to be donor-like)
EACORN.T	real	0.0	energy below E_C at which conduction band tail begins (eV)
EDCORN.T	real	0.0	energy above E_V at which valence band tail begins (eV)
CAP.T	real	1×10^{-14}	capture cross-section for holes in the conduction band tail (cm^2)
CAN.T	real	1×10^{-14}	capture cross-section for electrons in the conduction band tail (cm^2)
CDP.T	real	1×10^{-14}	capture cross-section for holes in the valence band tail (cm^2)
CDN.T	real	1×10^{-14}	capture cross-section for electrons in the valence band tail (cm^2)

LAYER (continued)

Parameter	Type	Default	Description
A0	real	0	radiative recombination coeff. (cm ³ /s)
AP	real	0	hole Auger coeff. (cm ⁶ /s)
AN	real	0	electron Auger coeff. (cm ⁶ /s)
TAUP.SHR	real(5)	1×10^6	SHR hole lifetime (s)
TAUN.SHR	real(5)	1×10^6	SHR electron lifetime (s)
ET.SHR	real(5)	0.0	SHR trap level wrt E _i (eV)
NTT.SHR	real(5)	0.0	density of SHR recomb centers (cm ⁻³) (donor-like, if NTT.SHR > 0) (acceptor-like, if NTT.SHR < 0)
SIG.SHR	real(5)	0.0	standard deviation (in eV) of gaussian SHR distribution
NTT.D	real	0.0	D level density (cm ⁻³)
SIG.D	real	0.0	standard deviation (in eV) of gaussian D level distribution
ET.D+ ETL.D+ ETR.D+	real	0.5	D ⁺ trap level wrt E _V (eV)
ET.D- ETL.D- ETR.D-	real	0.5	D ⁻ trap level wrt E _V (eV)
CP.D-	real	1×10^{-14}	D ⁻ → D ⁰ hole capture cross section (cm ²)
CP.D0	real	1×10^{-14}	D ⁰ → D ⁺ hole capture cross section (cm ²)
CN.D+	real	1×10^{-14}	D ⁺ → D ⁰ electron capture cross section (cm ²)
CN.D0	real	1×10^{-14}	D ⁰ → D ⁻ electron capture cross section (cm ²)

LAYER (continued)

Parameter	Type	Default	Description
EG1.OPT	real	-1.	optical bandgap (eV) (-1 sets EG1.OPT = EG)
EG1L.OPT			
EG1R.OPT			
C0.EG1	real	0.0	parameters used in optical absorption calculation
C1.EG1	real	0.0	
C2.EG1	real	0.0	
C3.EG1	real	0.0	
C4.EG1	real	0.0	
C5.EG1	real	0.0	
C6.EG1	real	0.0	
C7.EG1	real	0.0	
EG2.OPT	real	0.	optical bandgap (eV)
EG2L.OPT			
EG2R.OPT			
C0.EG2	real	0.0	parameters used in optical absorption calculation
C1.EG2	real	0.0	
C2.EG2	real	0.0	
C3.EG2	real	0.0	
C4.EG2	real	0.0	
C5.EG2	real	0.0	
C6.EG2	real	0.0	
C7.EG2	real	0.0	
EG3.OPT	real	0.0	optical bandgap (eV)
EG3L.OPT			
EG3R.OPT			
C0.EG3	real	0.0	parameters used in optical absorption calculation
C1.EG3	real	0.0	
C2.EG3	real	0.0	
C3.EG3	real	0.0	
C4.EG3	real	0.0	
C5.EG3	real	0.0	
C6.EG3	real	0.0	
C7.EG3	real	0.0	

LAYER (continued)

Parameter	Type	Default	Description
TAIL.OPT	logical	FALSE	if TRUE, absorption tail included
ALFC.OPT	real	0.0	parameters used in absorption
ZETA.OPT	real	0.0	tail calculation (see notes)
PH.OPT	logical	FALSE	if TRUE, phonon assisted absorption included
EP1.OPT	real	0.0	phonon assisted absorption
EP2.OPT	real	0.0	parameters (see notes)
ALF.OPT	real(6)		
MODEL	char		set parameters for specific materials =CIS.Cu- for Cu-poor CIS =CIS.Cu+ for Cu-rich CIS =CIS.st for Stoichometric CIS =i-CdS for intrinsic CdS =p-CdTe for p-type CdTe =n-CdTe for n-type CdTe =a-Si for α -Si

GENREC

Parameter	Type	Default	Description
GEN	char	DARK	sets generation rate options- DARK UNIFORM MONO (monochromatic) AM1P5G, etc. (see notes)
WL	real		wavelength for GEN=MONO (μm)
JINC	real	0.1	incident flux $\times q$ (A/cm^2) (GEN=MONO,UNIFORM, or spectral response (SR) option)
CONC	real	1.0	multiplication factor for illumination intensity
SHADOW	real	0.0	relative shadowing factor
ANGLE	real	0.0	angle of incidence
RBACK	real	0.0	relative back surface reflectance
RFRONT	real	0.0	relative internal front surface reflectance

I-V

Parameter	Type	Default	Description
VSTART	real	0.0	start voltage (V)
VSTOP	real	0.0	stop voltage (V)
DV	real	0.0	voltage increment (V)
V	real()	0.0	voltage (V)

SOLCELL

Parameter	Type	Default	Description
GFACTOR	integer	10	$10^{-\text{GFACTOR}}$ is the initial generation rate factor for stepping up the intensity
VSTART	real	0.0	start voltage (V)
DVSMALL	real	0.03	smallest voltage increment (V)
DVBIG	real	0.10	largest voltage increment (V)
VSMALL	real	100.	voltage at which DVBIG becomes DVSMALL
RSERIES	real	0.0	external series resistance ($\Omega\text{-cm}^2$)
RSHUNT	real	1×10^{51}	external shunt resistance ($\Omega\text{-cm}^2$)

SR

Parameter	Type	Default	Description
VA	real	0.0	applied voltage (V)
J(VA)	real	0.0	dark current at $V=VA$ (A/cm^2) $(JA(VA) > 0 \text{ if } VA > 0)$
WL	real()		wavelength (μm) defaults to WL=.34/.36/.38/.40/.44 .46/.48/.50/.52/.54/.56/.58/.60 .62/.64/.66/.68/.72/.76

IR

Parameter	Type	Default	Description
JINC	real	0.1	incident flux $\times q$ (A/cm ²)
VA	real	0.0	applied voltage (V)
XMIN	real	0.0	start position for impulse response (μm)
XMAX	real	-1.0	stop position for impulse response (μm) (defaults to device length)
DX	real	-1.0	step size for impulse generation (μm) (defaults to 50 steps)